A Ranking Stability Measure for Quantifying the Robustness of Anomaly Detection Methods

Lorenzo Perini[0000−0002−5929−9727] (✉), Connor Galvin, and
Vincent Vercruyssen[0000−0003−3645−3135] (✉)

DTAI Research Group & Leuven.AI, KU Leuven, Belgium
firstname.lastname@kuleuven.be

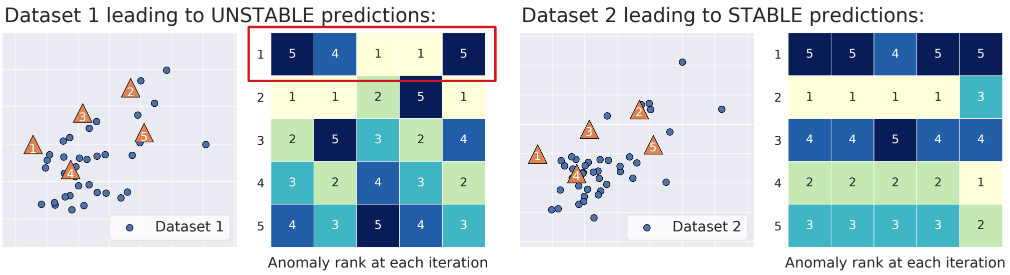
Abstract. Anomaly detection attempts to learn models from data that can detect anomalous examples in the data. However, naturally occurring variations in the data impact the model that is learned and thus which examples it will predict to be anomalies. Ideally, an anomaly detection method should be robust to such small changes in the data. Hence, this paper introduces a *ranking stability measure* that quantifies the robustness of any anomaly detector's predictions by looking at how consistently it ranks examples in terms of their anomalousness. Our experiments investigate the performance of this stability measure under different data perturbation schemes. In addition, they show how the stability measure can complement traditional anomaly detection performance measures, such as area under the ROC curve or average precision, to quantify the behaviour of different anomaly detection methods.

Keywords: Ranking Stability · Anomaly Detection · Classifier Trust.

1 Introduction

Anomaly detection attempts to find examples in a dataset that do not conform normal behaviour. It has many applications in areas such as fraud detection, medical disease detection, and cyber security [5]. Because anomalies are in nature unexpected and happen infrequently, the datasets observed in many anomaly detection tasks are especially prone to variation. Hypothetically, if we could collect a dataset multiple times, it would contain different anomalous and normal examples each time. Consequently, an anomaly detection model learned from the data, will each time be slightly different and make different predictions. It would be valuable to quantify just how consistent a detector's predictions are under variations in the training data. We refer to this as the detector's *stability*.

Anomaly detection is typically tackled from an unsupervised perspective because acquiring labels in real world use cases is expensive [17]. For instance, you will not simulate a medical error simply to have an example of anomalous behaviour. Given some training data, an anomaly detection model assigns an anomaly score to each example in the test set. Because the magnitude of this score conveys the anomalousness of an example, a ranking can be constructed over the test set examples from most to least anomalous. This helps the user to



1	5	4	1	1	5
2	1	1	2	5	1
3	2	5	3	2	4
4	3	2	4	3	2
5	4	3	5	4	3

1	5	5	4	5	5
2	1	1	1	1	3
3	4	4	5	4	4
4	2	2	2	2	1
5	3	3	3	3	2

Fig. 1: Illustration of how uncertainty in which training data are observed leads to instability in the rankings. For five iterations (columns), we take a random subset from each dataset (blue dots) to train an Lof model, predict the anomaly scores for the test set examples (orange triangles), and rank them from least anomalous (rank 1) to most anomalous (rank 5). Clearly, small perturbations of dataset 2 have a smaller effect on the consistency of Lof's predictions than small perturbations in dataset 1.

inspect the predictions in a structured way. However, user trust in the anomaly detector's predictions will quickly erode if retraining the model on a slightly different version of the training data yields different predictions (and thus a different ranking). Figure 1 illustrates this process on two toy datasets. Due to different examples being observed in each dataset, the anomaly detection model ranks the test set examples differently each time it is retrained on a subset of the dataset. If the user could quantify the *stability* of a model, it would help her decide which model to use or how to collect her data.

To the best of our knowledge, this paper is the first to contribute a *ranking stability measure* for quantifying the robustness of anomaly detection methods. The stability measure is constructed by retraining an anomaly detector multiple times on different subsets of the training data, each time constructing the ranking over the test set examples, and measuring the consistency between these rankings. The stability measure is computed in a completely unsupervised manner, unlike popular existing performance measures for anomaly detection that require labeled data, such as the area under the ROC curve or average precision [4]. In contrast, our stability measure captures a different aspect of a model's performance, namely its ability to consistently make the same predictions for a set of examples when the training data change.[1] We perform an extensive empirical evaluation of our stability measure and show that it indeed responds meaningfully to changes in the training data. Finally, we conduct a comparison between seven state-of-the-art anomaly detection methods in terms of the stability measure and traditional performance measures.

[1] We will experimentally validate this claim in Section 4 by measuring correlations between our stability measure and existing performance measures.

2 Related Work

Numerous anomaly detection methods have been developed during the past decades [4,6]. The focus of this paper, however, is not on any particular method, but on designing a way to measure the stability of a method. The most closely related work in this area is [14]. The authors develop ExCeeD, a method to estimate the confidence of any anomaly detector in its example-wise predictions. In contrast, our stability measure does not look at binary predictions but at how different examples are comparatively ranked by an anomaly detector, providing an *aggregate* picture of stability instead.

The idea of stability has sprung up in different areas of machine learning. In particular, [10] proposed a *point stability* measure with respect to clustering. It captures the idea that if two examples belong to the same cluster executed on some subset of the data, they do not necessarily belong to the same cluster if the full dataset is clustered. Similarly, we are interested in the stability of an example's ranking by an anomaly detector. Our measure, however, is not suitable for clustering, while the metrics developed in [10] cannot be applied to anomaly detection or rankings.

Social choice theory studies how individual opinions, preferences, or interests (i.e., rankings of items) can be combined to form a social consensus. Most work is being done on developing algorithms that can derive the consensus ranking from individual's rankings [1]. In addition, rank correlation and rank distance metrics have been developed to measure the agreement between a pair of orderings of items [7]. In contrast, the fundamental insight of our stability measure is that, within the context of anomaly detection, (small) changes in the rank position of an example near the top of the ranking should contribute more to the aggregate stability than (large) changes near the bottom of the ranking.

Ensemble methods for anomaly detection often make use of rank aggregation techniques to aggregate the predictions of the different ensemble members [19,18]. In contrast, we look at rank aggregation from a post-hoc evaluation perspective. Our stability measure captures consistency in the rankings, but cannot be used to make predictions. This is akin to the intuition behind internal evaluation measures for anomaly detection such as Ireos [12,13], where the goal is to evaluate the predictive performance of anomaly detectors without access to the ground truth. Our stability measure does not evaluate predictive performance, however, but rather robustness of the predictions.

3 Methodology

This paper tackles the following problem:

Given: a training dataset D_{train} and a test set D_{test}, an anomaly detection model h, and a contamination factor γ;

Design: a stability measure $\mathcal{S}_h$ that quantifies the ability of the model h to rank the examples in D_{test} consistently under variations in D_{train}.

A trained anomaly detection model h computes an anomaly score for each example in D_{test}. These scores can be used to create a *ranking* of the test examples from least to most anomalous. Our key insight is that an unstable model will not produce consistent rankings when retrained on different, uniformly sampled subsets of the training data: the same test set example will sometimes be ranked high and sometimes low. Thus, we can define a model's *stability* in terms of the examples' stabilities:

$$\mathcal{S}_h \coloneqq \frac{1}{n_t}\sum_{j=1}^{n_t} \mathcal{S}_{x_j}, \tag{1}$$

where, for each example x_j in the test set D_{test} of size n_t, $\mathcal{S}_{x_j}$ captures the consistency in its position in the ranking when retraining h multiple times on variations of D_{train}. We can now estimate the model stability $\mathcal{S}_h$ in three steps. First, we randomly draw subsets from the training set D_{train} to simulate slight changes in the set of available training examples. Each time, we retrain model h and construct a ranking over D_{test}. Second, we assign a stability score to each test set example by taking into account both the variance and the range of its normalized rank positions. Third, we aggregate the stability scores of all test set examples to obtain the model score $\mathcal{S}_h$.

3.1 Generating Anomaly Rankings

Our goal is to design a measure that captures an anomaly detector's consistency in ranking test examples under slight variations of the training data. To simulate these variations, we draw I different subsets D_i from D_{train} without replacement, with $i = 1, \dots, I$ and $|D_i|$ randomly selected as a percentage of $|D_{train}|$. Each time, we retrain h and use it to predict the anomaly score of each test set example. This results in I sets of scores

$$S^{(I)} = \{S^{(i)} \subseteq \mathbb{R}^{n_t} \colon i = 1, \dots, I\} = \{\{s_1^{(i)}, \dots, s_{n_t}^{(i)} \in \mathbb{R} \colon n_t \in \mathbb{N}\} \colon i = 1, \dots, I\},$$

where

$$s_j^{(i)} = h_{D_i}(x_j)$$

is the anomaly score of the example x_j through h when training on subset $D_i \subseteq D_{train}$, and n_t is the number of test set examples. Then, we define the *rank positions* of each example x_j as

$$r_j^{(I)} = \{r_j^{(i)} \in \{1, \dots, n_t\} \colon i = 1, \dots I\}, \tag{2}$$

where $r_j^{(i)}$ represents the position of the score s_j among the n_t scores when the examples are sorted from lowest anomaly score (position 1) to highest anomaly score (position n_t). We normalize the rank positions by dividing each $r_j^{(i)}$ by n_t. Thus, for any example $x_j \in D_{test}$, its *normalized* list of rank positions will be referred to as $r_j^{(I)}$. For instance, the normalized rank positions of example 1 in dataset 1 of Figure 1 (red box) are [1.0, 0.8, 0.2, 0.2, 1.0].

3.2 Example-Wise Stability Score

An example's *stability* quantifies the variation in its normalized rank positions. The most obvious way to do this is to measure the standard deviation of an example's rank positions. However, this does not reflect that some changes in ranking are intuitively more important than others. Whether two normal examples change position in the ranking is not so important as whether an anomaly suddenly ranks lower than a normal example. In other words, we care mostly about variations in the top part of the ranking that presumably contains the anomalies (or at least the examples that the model thinks are the anomalies). Knowing the proportion of anomalies in the test set, i.e., the contamination factor γ, we can consider all examples in the top $(1-\gamma)$ % of the ranking to be the anomalies, while the rest are the normals. Thus, for each example $x_j \in D_{test}$, we compute its stability score as:

$$\mathcal{S}_{x_j} = 1 - \frac{1}{Z}\left[\sqrt{\mathrm{Var}\left[r_j^{(I)}\right]} \times \omega\left(r_j^{(I)};\gamma\right)\right] \tag{3}$$

where the first multiplicative term is the standard deviation of an example's rank positions. This term has values in the range $[0, 0.5]$, as proven in Theorem 1 (see Appendix). The ω-term captures the intuition that an example's ranking should be considered more unstable if its rank positions change near the top as determined by γ. Finally, Z is a normalization constant (see Section 3.3).

To model the ω-term, we make use of a $Beta\,(\alpha,\beta)$ distribution defined over the range of all possible normalized rank positions (going from 0 to 1). By carefully setting the α and β parameters, the shape of the distribution can be tailored to our task. First, we set the parameters such that the *mode* of the distribution coincides with the threshold between predicted anomalies and normals:

$$\frac{\alpha-1}{\alpha+\beta-2} = 1-\gamma \implies \gamma\alpha - (1-\gamma)\beta = 2\gamma - 1. \tag{4}$$

Second, we require that ψ% of the mass of the Beta distribution falls within the interval $[1-2\gamma, 1]$. This has the intuitive interpretation that rank changes within this interval have more weight. Hyperparameter ψ is set by the user. To enforce these constraints, we solve the following optimization problem:

$$\min_{\alpha,\beta} \quad \left[(1-\psi) - \mathrm{F}_{\alpha,\beta}(1-2\gamma)\right]^2 \tag{5a}$$

$$\text{subject to} \quad \alpha \geq 1, \beta \geq 1, \tag{5b}$$

$$\gamma\alpha - (1-\gamma)\beta = 2\gamma - 1 \tag{5c}$$

where $\mathrm{F}_{\alpha,\beta}(1-2\gamma)$ is the cumulative density function of a Beta distribution governed by parameters α and β and evaluated at rank position $1-2\gamma$.

The key insight is that the area under the Beta distribution can capture the uncertainty caused by the spread of all possible rankings of an example:

$$\omega\left(r_j^{(I)};\gamma\right) = \int_{\min_i\left\{r_j^{(i)}\right\}}^{\max_i\left\{r_j^{(i)}\right\}} \frac{x^{\alpha-1}(1-x)^{\beta-1}}{\mathrm{B}\left(\alpha,\beta\right)}\,dx, \tag{6}$$

where $\mathrm{B}(\alpha, \beta)$ is the beta function. Thus, an example that has a large range of rankings, as indicated by $[\min_i\{r_j^{(i)}\}, \max_i\{r_j^{(i)}\}]$, or whose range is closer to the top of the ranking, is penalized more when computing its stability. If an example is ranked as both the most anomalous (position n_t) and least anomalous (position 1) when training h on different subsets, it gets the maximum penalty possible, which is 1 (the entire area under the Beta distribution).

3.3 The Model Stability Measure

The model stability measure $\mathcal{S}_h$ takes values in the range $[0, 1]$, where a score of 1 means that each of the I rankings is identical. We set the normalization constant Z of Equation 3 such that a stability score of 0 corresponds to a model that produces completely random rankings each time it is retrained. In fact, we assume that a model performing worse than randomly ranking the examples also gets a stability measure equal to 0, as we are not interested in measuring stability for such unstable scenarios. Thus, in a random scenario (i.e. worst case), the constant Z is equal to the standard deviation of a discrete uniform random variable, as shown in Theorem 2 (see Appendix).

The final stability measure for a model h is obtained by taking the average of the stability scores of all test set examples, as defined in Equation 1. The sample mean allows us to infer a unique stability measure for model h by uniformly weighting all the example-wise stability scores.

4 Experiments

In this section, we try to answer the following questions:[2]

Q1: Does the stability measure behave as we would expect it to?
Q2: How do the hyperparameters of the stability measure influence its value?
Q3: Can we use the stability measure to compare different anomaly detection algorithms, complementing traditional performance measures?

Data. For all experiments in this section, results are presented on 9 datasets that are commonly used in anomaly detection [4]. The datasets vary in number of samples, dimensionality, and γ.[3]

Anomaly Detectors. To test the stability measure, we use 7 well-known anomaly detectors: Lof [3], Knno [15], iForest [11], Hbos [8], Inne [2], Ocsvm [16], and Cblof [9].

4.1 Results Q1: Behaviour of the Stability Measure

To see whether the stability measure behaves as we would like it to, we test the following hypothesis: "an anomaly detector trained on subsequent *biased* subsets

[2] Code available at: `https://github.com/Lorenzo-Perini/StabilityRankings_AD`.

[3] Details of data: `https://www.dbs.ifi.lmu.de/research/outlier-evaluation/`

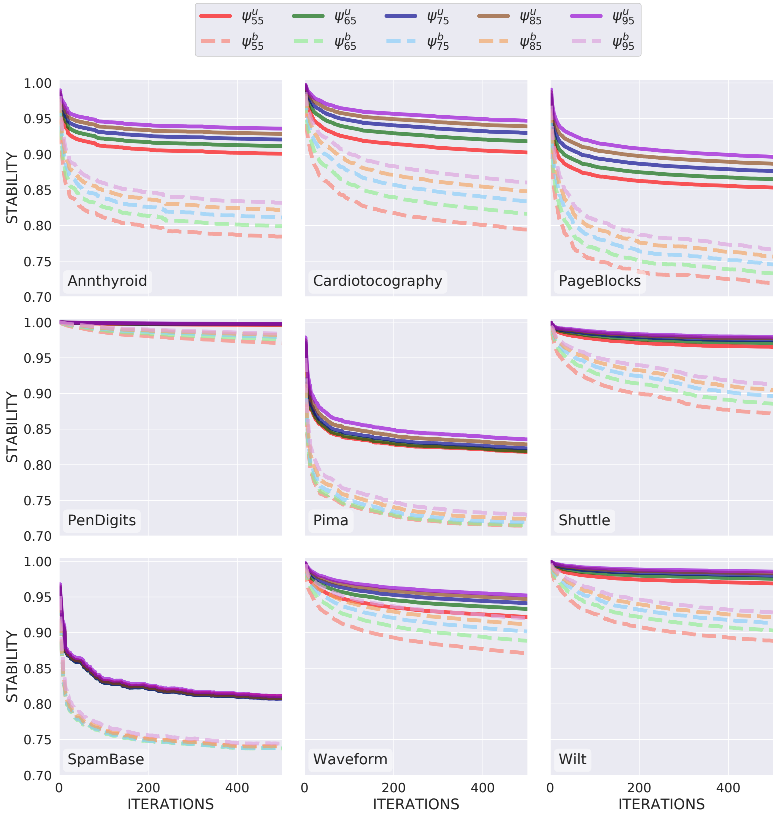


Fig. 2: The figure illustrates how the stability measure behaves under two different strategies to sample the I subsets: uniform sampling (continuous lines) and biased sampling (dashed lines). We vary both the number of iterations I (x-axis) and hyperparameter ψ (lines). Biased sampling of the subsets decreases the stability.

of a dataset, should have a lower stability than the same detector trained on *random* subsets of the same dataset.” To simulate this, we compute two versions of our stability measure for a given dataset: (1) a version where the I subsets are drawn uniformly from D_{train}; (2) a version where the subsets are drawn in a biased manner. In practice, we achieve the latter by first clustering D_{train} into 10 clusters and assigning a random weight to all instances in each cluster between every subset iteration. For a given dataset, we repeat this experiment for three anomaly detectors (Lof, Knno, and iForest) using 5-fold cross-validation for each detector, and report the average stability over all folds and detectors. We use different detectors to factor out the dependence on a single model.

Figure 2 shows the results for each of the 9 benchmark datasets. The plot also shows how the stability changes for values of hyperparameters I and ψ. The

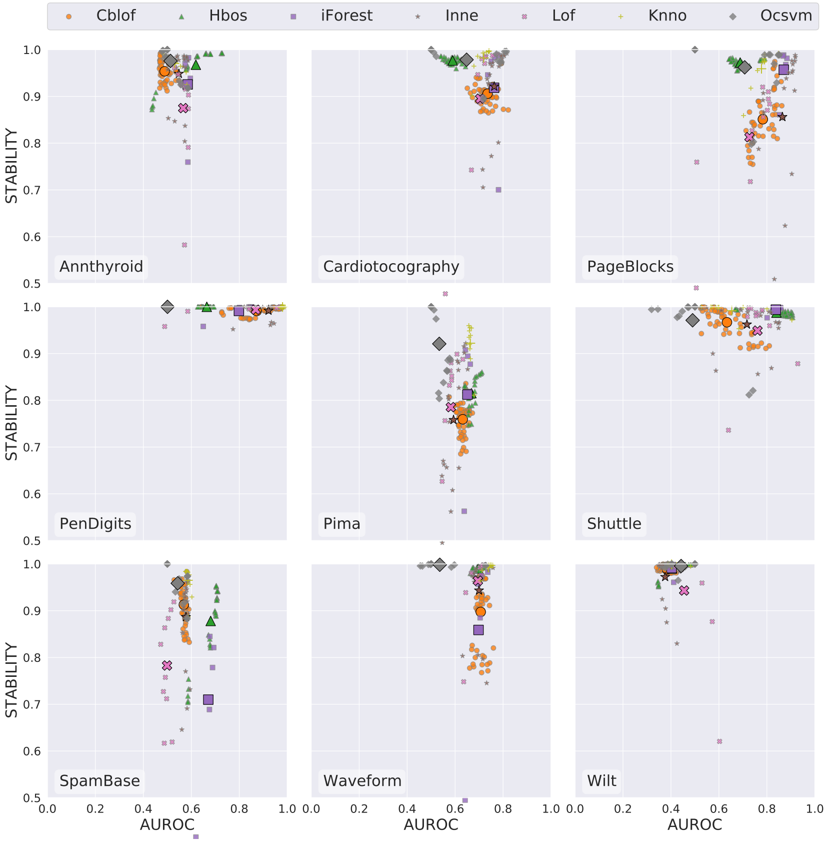


Fig. 3: Each dot of a certain color represents the stability and area under the ROC curve achieved by one of the 7 anomaly detectors with a given hyperparameter setting on the dataset. The large dots are the corresponding averages per method. The figure illustrates that the different detectors have varying stabilities.

results confirm our hypothesis: the stability of an anomaly detector trained on uniformly drawn subsets is always higher than that of the same detector trained on biased subsets. This provides evidence that our stability measure conforms our intuitions on how it should behave.

4.2 Results Q2: Hyperparameters

The stability measure has two hyperparameters: the number of iterations I and the shape of the Beta distribution ψ. Figure 2 shows how the stability measure changes for different values of these hyperparameters. For each of the 9 datasets, it seems that the stability measure converges after about 250 iterations. For most datasets, the value of ψ does not have a large impact as long as it is around 75%.

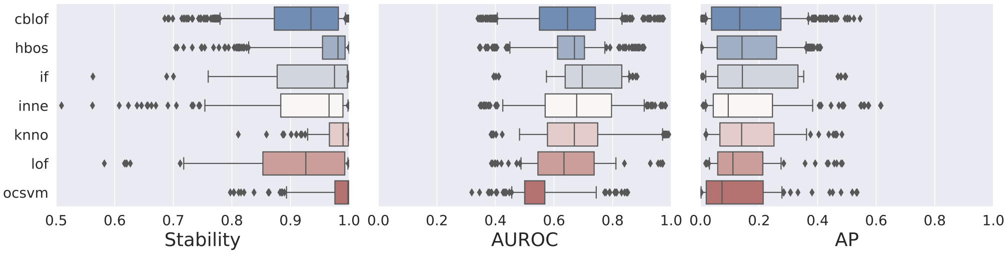


Fig. 4: Comparison of different anomaly detectors in terms of: stability, AUROC, and average precision. For each method and measure, a box plot shows the full range of results obtained over all benchmark datasets and hyperparameter settings.

4.3 Results Q3: Comparing Anomaly Detectors

We apply the 7 anomaly detection methods, each with a variety of hyperparameter settings, to the benchmark datasets and record our stability measure, the area under the ROC curve (AUROC), and the average precision. We compute them using 5-fold cross-validation. Figure 3 plots the results as a scatter plot where each dot represents both the stability and AUROC achieved by applying a method with a certain hyperparameter setting to the corresponding dataset. A large dot corresponds to the performance of a method averaged over all its hyperparameter settings. The figure allows us to compare different methods in terms of their stability. In all datasets, Ocsvm has on average the highest stability, meaning that the method is most robust to changes in the training data. The least stable method is Cblof, which might not be surprising given that it relies on a k-means clustering subroutine and changes in the data easily affect which clusters are found.

Figure 4 aggregates all results for each method and per measure (stability, AUROC, and average precision) over the entire benchmark. Both Knno and Ocsvm seem to be stable methods. Finally, the Pearson correlations between stability and AUROC and average precision are around -0.05 and -0.4 respectively, indicating that our stability measure captures a different aspect of performance than those traditionally used metrics.

5 Conclusion

We proposed a method to quantify the robustness of anomaly detectors by measuring the ranking stability under slight variations of the training data. The method estimates the stability in three steps. First, we simulate perturbations in the training set by drawing i.i.d. subsets. Second, we estimate the example-wise stability score by taking into account both the standard deviation of normalized rankings and the area under a Beta distribution. Third, we derive the stability measure by averaging the normalized stability score. The experiments show that the stability measure can meaningfully capture ranking variations and be a valid alternative to traditional performance measures to quantify the behaviour of different anomaly detectors.

Acknowledgements

This work is supported by the Flemish government under the "*Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen*" programme (LP, VV).

References

1. Amodio, S., D'Ambrosio, A., Siciliano, R.: Accurate algorithms for identifying the median ranking when dealing with weak and partial rankings under the kemeny axiomatic approach. European Journal of Operational Research **249**(2), 667–676 (2016)
2. Bandaragoda, T.R., Ting, K.M., Albrecht, D., Liu, F.T., Zhu, Y., Wells, J.R.: Isolation-based anomaly detection using nearest-neighbor ensembles. Computational Intelligence **34**(4), 968–998 (2018)
3. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: Lof: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data. pp. 93–104 (2000)
4. Campos, G.O., Zimek, A., Sander, J., Campello, R.J., Micenková, B., Schubert, E., Assent, I., Houle, M.E.: On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. Data Mining and Knowledge Discovery **30**(4), 891–927 (2016)
5. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM computing surveys (CSUR) **41**(3), 1–58 (2009)
6. Domingues, R., Filippone, M., Michiardi, P., Zouaoui, J.: A comparative evaluation of outlier detection algorithms: Experiments and analyses. Pattern Recognition **74**, 406–421 (2018)
7. Emond, E.J., Mason, D.W.: A new rank correlation coefficient with application to the consensus ranking problem. Journal of Multi-Criteria Decision Analysis **11**(1), 17–28 (2002)
8. Goldstein, M., Dengel, A.: Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. KI-2012: Poster and Demo Track pp. 59–63 (2012)
9. He, Z., Xu, X., Deng, S.: Discovering cluster-based local outliers. Pattern Recognition Letters **24**(9-10), 1641–1650 (2003)
10. Höppner, F., Jahnke, M.: Holistic assessment of structure discovery capabilities of clustering algorithms. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 223–239. Springer (2019)
11. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation-based anomaly detection. ACM Transactions on Knowledge Discovery from Data (TKDD) **6**(1), 1–39 (2012)
12. Marques, H.O., Campello, R.J.G.B., Zimek, A., Sander, J.: On the internal evaluation of unsupervised outlier detection. In: Proceedings of the 27th International Conference on Scientific and Statistical Database Management (2015)
13. Marques, H.O., Campello, R.J., Sander, J., Zimek, A.: Internal evaluation of unsupervised outlier detection. ACM Transactions on Knowledge Discovery from Data (TKDD) **14**(4), 1–42 (2020)

14. Perini, L., Vercruyssen, V., Davis, J.: Quantifying the confidence of anomaly detectors in their example-wise predictions. In: The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases. Springer Verlag (2020)
15. Ramaswamy, S., Rastogi, R., Shim, K.: Efficient algorithms for mining outliers from large data sets. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data. pp. 427–438 (2000)
16. Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C.: Estimating the support of a high-dimensional distribution. Neural computation **13**(7), 1443–1471 (2001)
17. Vercruyssen, V., Wannes, M., Gust, V., Koen, M., Ruben, B., Jesse, D.: Semi-supervised anomaly detection with an application to water analytics. In: Proceedings of the IEEE International Conference on Data Mining. (2018)
18. Zimek, A., Campello, R.J., Sander, J.: Data perturbation for outlier detection ensembles. In: Proceedings of the 26th International Conference on Scientific and Statistical Database Management. pp. 1–12 (2014)
19. Zimek, A., Gaudet, M., Campello, R.J., Sander, J.: Subsampling for efficient and effective unsupervised outlier detection ensembles. In: Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 428–436 (2013)

6 Appendix

Theorem 1. *Let $(\Omega, \Im, \mathbb{P})$ be a probability space, where Ω is a set, $\Im$ represents a σ-algebra over Ω and $\mathbb{P}$ is a probability measure. Assume that $X \in L^2$ is a real random variable (continuous or discrete), such that $X\colon \Omega \to \mathcal{V} \subseteq [0,1]$. Then,*

$$0 \leq \mathrm{Var}\left[X\right] \leq \frac{1}{4}.$$

Proof. Assuming that $0 \leq X \leq 1$,

$$\mathrm{Var}\left[X\right] = \mathbb{E}\left[X^2\right] - \mathbb{E}\left[X\right]^2 \leq \mathbb{E}\left[X\right] - \mathbb{E}\left[X\right]^2,$$

where the inequality holds because $0 \leq X \leq 1$. Then,

$$\begin{aligned}\mathrm{Var}\left[X\right] &\leq \mathbb{E}\left[X\right] - \mathbb{E}\left[X\right]^2 = -\Big(\mathbb{E}\left[X\right]^2 - \mathbb{E}\left[X\right]\Big)\\ &= -\left[\left(\mathbb{E}[X] - \frac{1}{2}\right)^2 - \frac{1}{4}\right] = \frac{1}{4} - \left(\mathbb{E}[X] - \frac{1}{2}\right)^2 \leq \frac{1}{4}.\end{aligned}$$

□

Theorem 2. *Let X be a discrete random variable over the probability space $(\Omega, \Im, \mathbb{P})$, where $\Omega = \{1, 2, \ldots, n_t\}$, $n_t \in \mathbb{N}$, $\Im$ a $\sigma-$algebra over Ω and $\mathbb{P}$ a probability measure. Assume that X follows a discrete uniform distribution. Then, the random variable $\frac{X}{n_t}$ has mean and variance, respectively, equal to*

$$\mathbb{E}\left[\frac{X}{n_t}\right] = \frac{n_t + 1}{2n_t}, \quad \mathrm{Var}\left[\frac{X}{n_t}\right] = \frac{(n_t+1)(n_t-1)}{12n_t^2} \tag{7}$$

Proof. We can directly compute the mean and the variance of $\frac{X}{n_t}$ by using the properties of random variables. First, we compute first and second moment of a discrete uniform random variables:

$$\begin{aligned} \mathbb{E}\left[X\right] &= \sum_{j=1}^{v} x_j \cdot \mathbb{P}(X = x_j) = \frac{1}{n_t}\sum_{j=1}^{n_t} j = \frac{n_t + 1}{2} \\ \mathbb{E}\left[X^2\right] &= \sum_{j=1}^{n_t} x_j^2 \cdot \mathbb{P}(X = x_j) = \frac{1}{n_t}\sum_{j=1}^{n_t} i^2 = \frac{(n_t + 1)(2n_t + 1)}{6}. \end{aligned} \tag{8}$$

Second, we achieve the goal as follows:

$$\begin{aligned} \mathbb{E}\left[\frac{X}{n_t}\right] &= \frac{1}{n_t}\mathbb{E}[X] = \frac{n_t + 1}{2n_t}; \\ \mathrm{Var}\left[\frac{X}{n_t}\right] &= \frac{1}{n_t^2}\mathrm{Var}\left[X\right] = \frac{1}{n_t^2}\left[\mathbb{E}\left[X^2\right] - \mathbb{E}\left[X\right]^2\right] = \frac{(n_t + 1)(n_t - 1)}{12n_t^2}. \end{aligned} \tag{9}$$

□